AI Research Labs

Deep Learning Foundations · Practice test 2 · 15 questions

# The Neuron and Linear Separability

**How to use this test.** Answer all 15 questions without notes, then mark yourself with the answer key at the back. Each question has exactly one best answer. Score 75% or more and you've mastered the material; below that, reread the lessons for the questions you missed.

## 2.1 The Artificial Neuron

Q1 calculate A neuron has $w = (0.5, 1.5)$, $b = -0.2$ and input $x = (2, -1)$. What is its pre-activation $z$?

A $+2.3$
B $-0.7$
C $+0.3$
D $-0.5$

Q2 calculate The same neuron (pre-activation $z = -0.7$) uses a ReLU activation. What does it output?

A −1
B 0
C 0.332
D −0.7

Q3 concept What does the bias $b$ of a neuron do?

A It scales every input by the same constant factor
B It measures the systematic error of the neuron's predictions
C It shifts the threshold, independently of the inputs
D It is the activation function applied after the weighted sum

Q4 interpret One input of a trained neuron has weight $-2.4$; another has weight $0.01$. What do these tell you?

A The first input is unimportant because its weight is negative
B Both inputs raise the output, since weights only set the speed of learning
C The second input matters most because its weight is the smallest
D The first input strongly lowers the output; the second barely matters

Q5 concept Why can the bias be written as an extra weight $w_0$?

A Add a constant input $x_0 = 1$, so $b = w_0 \cdot 1$
B Because the bias must be zero for gradient descent to work
C Because the bias is always equal to the first weight
D Because the activation absorbs the bias into its formula

## 2.2 Decision Boundaries and Linear Separability

Q6 `concept` For a neuron with a sign activation, what is its decision boundary?

- A The curve where the activation's derivative is largest
- B The set of inputs where the output is exactly 1
- C The set of inputs where $w^\top x + b = 0$
- D The line through the origin perpendicular to $b$

Q7 `calculate` A neuron with weights $(1, 1)$, bias $-1.5$ and a sign activation receives the four binary inputs. Which function does it compute?

- A AND
- B NAND
- C OR
- D XOR

Q8 `concept` Which direction does the weight vector $w$ point relative to the decision boundary?

- A Through the origin, whatever the value of the bias
- B Along it, from one end of the boundary to the other
- C Perpendicular to it, towards the negative side
- D Perpendicular to it, towards the positive side

Q9 `apply` Which of these datasets is linearly separable?

- A One class inside a circle, the other in a ring around it
- B XOR on two binary inputs
- C OR on two binary inputs
- D Two interleaved spirals, one for each of the two classes

Q10 `interpret` You keep $w = (1, 1)$ but change the bias from $-1.5$ to $-0.5$. What happens to the boundary?

- A It becomes curved, because the bias is now smaller
- B It rotates by 45°, turning AND into XOR
- C It slides parallel to itself, turning AND into OR
- D It stays where it is, since only weights move it

## 2.3 The XOR Problem

Q11 `concept` In the proof that one neuron can't compute XOR, what contradiction do the four inequalities produce?

- A The output must be both 0.5 and 1 at the same input
- B $w_1 + w_2 + 2b$ must be both positive and negative
- C Both of the weights must be negative and also positive
- D The bias $b$ must be both zero and also non-zero

Q12 `interpret` You train a single sigmoid neuron on XOR with MSE. The loss stalls at 0.25 and every output is about 0.5. What does this mean?

A The learning rate is too small; train much longer and it will solve XOR
B The training data must be mislabelled, since XOR is linearly separable
C It has found the best a straight boundary can do: predict the mean
D The weights have exploded, so the sigmoid outputs saturate at 0.5

Q13 `calculate` XOR can be written as $x_1 + x_2 - 2x_1x_2$. What does it give for $(1, 1)$?

A 2
B 0
C 1
D $-2$

Q14 `apply` You add the product $x_1x_2$ as a third input to a single neuron. Can it now compute XOR?

A Only if the activation function is also changed from sign to ReLU
B Yes: XOR is linear in $(x_1, x_2, x_1x_2)$
C No: a single neuron can never compute XOR, whatever inputs it is given
D Only with a hidden layer added after the new product feature

Q15 `concept` Why is the XOR result described as a proof rather than an experimental finding?

A It depends on the particular learning rate that was used in training
B It was verified by training many thousands of networks on XOR
C It shows no weights at all can work, not just that training failed
D It holds only for the sigmoid activation, not for any other

# Answer key

Each answer comes with the reasoning, including why the most tempting wrong option is wrong.

**Q1** **B** · lesson 2.1

$z = 0.5 \times 2 + 1.5 \times (-1) - 0.2 = 1 - 1.5 - 0.2 = -0.7$. $+0.3$ adds the bias with the wrong sign, $-0.5$ forgets it, and $2.3$ ignores the negative input.

**Q2** **B** · lesson 2.1

ReLU is $\max(0, z)$, and $z$ is negative, so the output is 0. $-0.7$ would be a linear activation, 0.332 the sigmoid of $-0.7$, and $-1$ the sign function.

**Q3** **C** · lesson 2.1

The bias is added to the weighted sum, moving where the neuron switches on; equivalently it's a weight on a constant input of 1. It doesn't scale inputs, it isn't the "bias" of bias–variance, and it isn't the activation.

**Q4** **D** · lesson 2.1

A large negative weight means more of that input pushes the pre-activation down strongly. A weight near zero means the input has almost no influence. The sign gives direction and the size gives strength.

**Q5** **A** · lesson 2.1

With an extra input fixed at 1, the term $w_0 x_0 = w_0$ plays the role of the bias, and the neuron becomes $f\left(\sum_{i=0}^{n} w_i x_i\right)$. The bias isn't tied to the first weight, needn't be zero, and isn't part of the activation.

**Q6** **C** · lesson 2.2

The neuron switches between $-1$ and $+1$ where its pre-activation changes sign, i.e. on the hyperplane $w^\top x + b = 0$. The bias moves it off the origin, and $w$ (not $b$) is perpendicular to it.

**Q7** **A** · lesson 2.2

$x_1 + x_2 - 1.5$ is positive only for $(1, 1)$, where it equals 0.5. That's AND. With bias $-0.5$ it would be OR; XOR can't be computed by one neuron; NAND is the negation of AND.

**Q8** **D** · lesson 2.2

For two points on the boundary, $w^\top (x_a - x_b) = 0$, so $w$ is normal to every direction along it. Moving along $w$ increases $z$, so $w$ points towards the side classified positive.

**Q9** **C** · lesson 2.2

A single line separates $(0, 0)$ from the three OR-positive points. XOR, a circle inside a ring, and interleaved spirals all need curved or multi-part boundaries.

**Q10** **C** · lesson 2.2

The weights fix the boundary's orientation; the bias only moves it. The line $x_1 + x_2 = 1.5$ shifts to $x_1 + x_2 = 0.5$, which now puts $(1, 0)$ and $(0, 1)$ on the positive side: OR. A linear neuron's boundary is always straight.

**Q11** **B** · lesson 2.3

The positive cases give $w_1 + b > 0$ and $w_2 + b > 0$, which add to $w_1 + w_2 + 2b > 0$. The negative cases give $b < 0$ and $w_1 + w_2 + b < 0$, which add to $w_1 + w_2 + 2b < 0$. Impossible.

**Q12** **C** · lesson 2.3

No straight boundary separates XOR, so the MSE-optimal answer is the average target 0.5 everywhere, with loss $\frac{1}{4}\sum(0.5 - y)^2 = 0.25$. More training can't help, saturated sigmoids output 0 or 1 (not 0.5), and XOR is not linearly separable.

Q13 **B** · lesson 2.3

$1 + 1 - 2(1)(1) = 0$, matching XOR(1, 1) = 0. The $-2x_1x_2$ term cancels the double count when both inputs are on; without it you'd get 2.

Q14 **B** · lesson 2.3

With weights $(1, 1, -2)$ on $(x_1, x_2, x_1x_2)$ and a suitable bias, one neuron separates XOR. The impossibility proof is for the original two inputs. The catch is that someone had to design the feature; a hidden layer learns such features.

Q15 **C** · lesson 2.3

The inequalities rule out every possible $w_1, w_2, b$, so no training method or setting can succeed. A failed experiment could be bad luck or a bad learning rate; a contradiction can't. It holds for any linear decision boundary.